
Lecture 8: Domain Modelling: Scripts and Human Behaviour

[Domain Modelling]

Scripts

For examples of narratives of domain scripts we refer to appendix Examples 3–5 (Slides 974–983).

Characterisation 69 (Domain Script) By a **domain script** we mean

- the structured wording of a rule or a regulation
- that has legally binding power,
- that is, which may be contested in a court of law

.



Domain Modelling, Scripts

Analysis of Examples

1. Bus [Train] Timetables (Schedules):

Slide 974

- The bus/train timetable is informally sketched.
- A later part of this lecture will elaborate, and formalise, this timetable example.
- In addition that part will relate timetables to the underlying net of to the resulting and possible traffics.
- A timetable script thus can be given several pragmatics:
 - ★ (i-ii) a, perhaps not exactly legally binding, contract between the bus/train operator and the passengers, as well as a contract between the bus/train operator and the public authorities;
 - ★ (iii) a particular timetable (considered as syntax) semantically denotes a possibly infinite set of bus/train traffics, each of which runs to schedule; and
 - ★ (iv) a script, to be followed by the drivers/train engine men, guiding these in the bus/train journey (to speed up or slow down, etc.).

[[Domain Modelling](#), [Scripts](#), [Analysis of Examples](#)]

2. Aircraft Flight Simulator Script:

Slides 976–979

- The example script is from a specific aircraft simulator demo. It has been abstracted a bit from the real case script.
- You may think of the example script being partly “programmed” into the flight simulator which is a reactive system awaiting pilot trainee actions and reactions.
- As you note, it is quite detailed. It mentions many phenomena and concepts of aircraft flights:
 - ★ entities (simple as well as behavioural),
 - ★ operations,
 - ★ events,
 - ★ and itself prescribes a behaviour.
- You may additionally think of the example script as also (in addition to the flight simulator hardware and software) “scripting” the pilot trainee.
- Thus a specific script, for example, denotes an infinity of actual behaviours of pilot trainees working in conjunction with flight simulators.

[Domain Modelling, Scripts, Analysis of Examples]

3. Bill of Lading:

Slides 980–983

- The bill of lading is also a script, but it is quite different from the previous two examples.
- It only very, very loosely hints at transport behaviours.
- Whereas it certainly puts some constraints on freight transport.
- The bill of lading script is a legal instrument which
 - ★ entitles the consignee to receive the freight at the destination harbour;
 - ★ stipulates, in the closing “conditions” item, legal protection of the two parties to the contract;
 - ★ etcetera.

Domain Modelling, Scripts

Licenses

License:

a right or permission
granted in accordance with law
by a competent authority

- to engage in some business or occupation,
- to do some act, or
- to engage in some transaction

which but for such license would be unlawful

Merriam Webster On-line

[Domain Modelling, Scripts, Licenses]

- The concepts of licenses and licensing express relations between
 - ★ *actors* (licensors (the authority) and licensees),
 - ★ *simple entities* (artistic works, hospital patients, public administration and citizen documents) and
 - ★ *operations* (on simple entities), and as performed by actors.
- By issuing a license to a licensee, a licensor wishes to express and enforce certain permissions and obligations:
 - ★ which operations
 - ★ on which entities
 - ★ the licensee is allowed (is licensed, is permitted) to perform.
- As such a license denotes a possibly infinite set of allowable behaviours.

[Domain Modelling, Scripts, Licenses]

- We shall consider three kinds of entities:
 - ★ (i) digital recordings of artistic and intellectual nature:
 - ◇ music, movies, readings (“audio books”), and the like,
 - ★ (ii) patients in a hospital
 - ◇ as represented also by their patient medical records, and
 - ★ (iii) documents related to public government.

[Domain Modelling, Scripts, Licenses]

- The *permissions* and *obligations* issues are,
 - ★ (i) for the owner (agent) of some intellectual property to be paid (i.e., an *obligation*) by users when they perform *permitted* operations (rendering, copying, editing, sub-licensing) on their works;
 - ★ (ii) for the patient to be professionally treated — by medical staff who are basically *obliged* to try to cure the patient; and
 - ★ (iii) for public administrators and citizens to enjoy good governance: transparency in law making (national parliaments and local prefectures and city councils), in law enforcement (i.e., the daily administration of laws), and law interpretation (the judiciary) — by agents who are basically *obliged* to produce certain documents while being *permitted* to consult (i.e., read, perhaps copy) other documents.

[Domain Modelling, Scripts, Licenses]

- In this section we shall rough-sketch-describe pragmatic aspects of the three domains of
 - ★ (1) production, distribution and consumption of artistic works,
 - ★ (2) the hospitalisation of patient, i.e., hospital health care and
 - ★ (3) the handling of law-based document in public government.
- The emphasis is on the pragmatics of the terms,
- i.e., the language used in these three domains.

Domain Modelling, Scripts

The Performing Arts: Producers and Consumers

- The intrinsic entities of the performing arts are the artistic works:
 - ★ drama or opera performances,
 - ★ music performances,
 - ★ readings of poems, short stories, novels, or jokes,
 - ★ movies,
 - ★ documentaries, newsreels, etc.
- We shall limit our span to the scope of electronic renditions of these artistic works:
 - ★ videos, CDs or other.
- In this talk we shall not touch upon the technical issues of “downloading” (whether ”streaming” or copying, or other).

[[Domain Modelling](#), [Scripts](#), [The Performing Arts: Producers and Consumers](#)]

● Operations on Digital Works ●

- For a consumer to be able to enjoy these works that consumer must (normally first) usually “buy a ticket” to their performances.
- The consumer, i.e., the theatre, opera, concert, etc., “goer” (usually)
 - ★ cannot copy the performance (e.g., “tape it”),
 - ★ let alone edit such copies of performances.
- In the context of electronic, i.e., digital renditions of these performances the above “cannots” take on a new meaning.
 - ★ The consumer may copy digital recordings,
 - ★ may edit these,
 - ★ and may further pass on such copies or editions to others.

[**Domain Modelling**, **Scripts**, **Licenses**, The Performing Arts: Producers and Consumers, *Operations on Digital Works*]

[**The Performing Arts: Producers and Consumers**, *Operations on Digital Works*]

- To do so, while protecting the rights of the producers (owners, performers),
 - ★ the consumer requests permission to have the digital works transferred (“downloaded”) from the owner/producer to the consumer,
 - ★ so that the consumer
 - ◇ can **render** (“play”) these works on own rendering devices (CD, DVD, etc., players),
 - ◇ possibly can **copy** all or parts of them,
 - ◇ then possibly can **edit** all or parts of the copies, and,
 - ◇ finally, possibly can further **license** these “edited” versions to other consumers subject to payments to “original” licensor.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers]

● License Agreement and Obligation ●

- To be able to obtain these permissions
 - ★ the user agrees with the wording of some license
 - ★ and pays for the rights to operate on the digital works.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers]

● Two Assumptions ●

- Two, related assumptions underlie the pragmatics of the electronics of the artistic works.
 - ★ The first assumption is that
 - ◇ the format, the electronic representation of the artistic works is proprietary,
 - ◇ that is, that the producer still owns that format.
 - ★ Either the format
 - ◇ is publicly known
 - ◇ or it is not, that is, it is somehow “secret”.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, Two Assumptions]

- In either case we “derive” the second assumption (from the fulfilment of the first).
 - ★ The second assumption is that the consumer is not allowed to, or cannot operate³ on the works by own means (software, machines).
- The second assumption implies that
 - ★ acceptance of a license
 - ★ results in the consumer receiving software
 - ★ that supports the consumer in performing all operations on
 - ★ licensed works, their copies and edited versions:
 - ◇ rendering,
 - ◇ copying,
 - ◇ editing and
 - ◇ sub-licensing.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers]

● Protection of the Artistic Electronic Works ●

- The issue now is:
 - ★ how to protect the intellectual property
 - ★ (i.e., artistic) and financial (exploitation) rights
 - ★ of the owners of the possibly rendered, copied and edited works,
 - ★ both when, and when not further distributed.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers]

• A License Language •

type

0. L_n, N_m, W, S, V
1. $L = L_n \times Lic$
2. $Lic == mkLic(licensor:N_m, licensee:N_m, work:W, cmds:Cmd\text{-}set)$
3. $Cmd == Rndr \mid Copy \mid Edit \mid RdMe \mid SuLi$
4. $Rndr = mkRndr(vw:(V \mid "work"), sl:S^*)$
5. $Copy = mkCopy(fvw:(V \mid "work"), sl:S^*, tv:V)$
6. $Edit = mkEdit(fvw:(V \mid "work"), sl:S^*, tv:V)$
7. $RdMe = "readme"$
8. $SuLi = mkSuLi(cs:Cmd\text{-}set, work:V)$

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (0.) Licenses

- ★ are given names,

- ◇ ln:Ln,

- ◇ so are actors (owners, licensors, and users, licensees), nn:Nm.

- ★ By w:W we mean a (net) reference to (a name of) the downloaded possibly segmented artistic work being licensed,

- ★ where segments are named (s:S), that is, s:S is a selector

- ◇ to either a segment of a downloaded work or

- ◇ to a segment of a copied and or and edited work.

[**Domain Modelling**, **Scripts**, **The Performing Arts: Producers and Consumers**, **A License Language**]

- (1.) Every license (lic:Lic) has a unique name (ln:Ln).
- (2.) A license (lic:Lic) contains four parts:
 - ★ the name of the licensor,
 - ★ the name of the licensee,
 - ★ a reference to (the name of) the work,
 - ★ a set of commands (that may be permitted to be performed on the work).

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (3.) A command is
 - ★ either a **render**, a **copy** or an **edit** or a **readme** command,
 - ★ or a sub-licensing (**sub-license**)
command.

[[Domain Modelling](#), [Scripts](#), [The Performing Arts: Producers and Consumers](#), [An Artistic Digital Rights License Language](#)]

- (4.–6.) The render, copy and edit commands are each “decorated” with an ordered list of selectors (i.e., selector names) and a (work) variable name.
 - ★ The license command
$$\mathbf{copy} \langle s_1, s_2, s_7 \rangle v$$
 - ★ means that the licensed work, ω , may have its sections s_1 , s_2 and s_7 copied, in that sequence, into a new variable named v ,
 - ★ Other copy commands may specify other sequences.
- Similarly for render and edit commands.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (7.) The "readme" license command,
 - ★ in a license, \ln , referring, by means of w , to work ω ,
 - ★ somehow displays a graphical/textual "image" of, that is, information about ω .
 - ★ We do not here bother to detail what kind of information may be so displayed.
 - ★ But you may think of the following display information
 - ◇ names of artistic work, artists, authors, etc.,
 - ◇ names and details about licensed commands,
 - ◇ a table of fees for performing respective licensed commands,
 - ◇ etcetera.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (8.) The license command

schema: **license** cmd1,cmd2,...,cmdn **on work** v

formal: mkSuLi({cmd1,cmd2,...,cmdn},v)

means

- ★ that the licensee is allowed to act as a licensor,
 - ★ to name sub-licensees (that is, licensees) freely,
 - ★ to select only a (possibly full) subset of the sub-licensed commands (that are listed) for the sub-licensee to enjoy.
- The license need thus not mention the name(s) of the possible sub-licensees.
 - ★ But one could design a license language, i.e., modify the present one to reflect such constraints.
 - The license also do not mention the payment fee component.
 - ★ As we shall see under licensor actions such a function will eventually be inserted.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- A license licenses the licensee to render, copy, edit and license (possibly the results of editing) any selection of downloaded works.
- In any order — but see below — and any number of times.
- For every time any of these operations take place payment according to the payment function occurs (that can be inspected by means of the **read license** command).
- The user can render the downloaded work and can render copies of the work as well as edited versions of these. Edited versions are given own names.
- Editing is always of copied versions.
- Copying is either of downloaded or of copied or edited versions.
- This does not transpire from the license syntax
 - ★ but is expressed by the licensee, see below,
 - ★ and can be checked and duly paid for according to the payment function.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- The payment function is considered a partial function of the selections of the work being licensed.
- Please recall that licensed works are proprietary.
 - ★ Either the work format is known, or it is not supposed to be known,
 - ★ In any case,
 - ◇ the rendering, editing, copying and the license-“assembling” (more later) functions
 - ◇ are part of the license and the licensed work
 - ◇ and are also assumed to be proprietary.
 - ★ Thus the licensee is not allowed to and may not be able to use own software for rendering, editing, copying and license assemblage.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- Licenses specify sets of permitted actions.
- Licenses do not normally mandate specific sequences of actions.
- Of course, the licensee, assumed to be an un-cloned human, can only perform one action at a time.
- So licensed actions are carried out sequentially.
- The order is not prescribed, but is decided upon by the licensee.
- Of course, some actions must precede other actions.
 - ★ Licensees must copy before they can edit,
 - ★ and they usually must edit some copied work before they can sub-license it.
 - ★ But the latter is strictly speaking not necessary.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

type

5. V

6. Act = Ln × (Rndr|Copy|Edit|License)

7. Rndr == mkR(sel:S*,wrk:(W|V))

8. Copy == mkC(sel:S*,wrk:(W|V),into:V)

9. Edit == mkE(wrks:V*,into:V)

10. License == mkL(ln:Ln,licensee:Nm,wrk:V,cmds:Cmd-**set**,fees:PF)

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (5.) By V we mean the name of a variable in the users own storage into which downloaded works can be copied (now becoming a local work).
 - ★ The variables are not declared.
 - ★ They become defined when the licensee names them in a copy command.
 - ★ They can be overwritten.
 - ★ No type system is suggested.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (6.) Every action of a licensee is tagged by the name of a relevant license.
 - ★ If the action is not authorised by the named license then it is rejected.
 - ★ Render and copy actions mention a specific sequence of selectors.
 - ★ If this sequence is not an allowed (a licensed) one, then the action is rejected.
 - ★ (Notice that the license may authorise a specific action, *a* with different sets of sequences of selectors — thus allowing for a variety of possibilities as well as constraints.)

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (7.) The licensee, having now received a license,
 - ★ can **render** selections
 - ◇ of the licensed work,
 - ◇ or of copied
 - ◇ and/or edited versionsof the licensed work.
 - ★ No reference is made to the payment function.
 - ◇ When rendering the semantics is that this function is invoked and duly applied.
 - ◇ That is, render payments are automatically made: subtracted from the licensee's account and forwarded to the licensor.

[**Domain Modelling**, **Scripts**, **The Performing Arts: Producers and Consumers**, **A License Language**]

- (8.) The licensee
 - ★ can **copy** selections
 - ◇ of the licensed work,
 - ◇ or of previously copied
 - ◇ and/or edited versionsof the licensed work.
 - ★ The licensee identifies a name for the local storage file where the copy will be kept.
 - ★ No reference is made to the payment function.
 - ◇ When copying the semantics is that this function is invoked and duly applied.
 - ◇ That is, copy payments are automatically made: subtracted from the licensee's account and forwarded to the licensor.

[Domain Modelling, Scripts, The Performing Arts: Producers and Consumers, A License Language]

- (9.) The licensee
 - ★ can **edit** selections
 - ◇ of the licensed work,
 - ◇ or of copied
 - ◇ and/or previously edited versions of the licensed work.
 - ★ The licensee identifies a name for the local storage file where the new edited version will be kept.
 - ★ The result of editing is a new work.

[Domain Modelling, Scripts, Licenses, Pragmatics of The Three License Languages]

[*The Performing Arts: Producers and Consumers, An Artistic Digital Rights License Language*]

- ★ No reference is made to the **payment** function.
 - ◇ When copying the semantics is that this function is invoked and duly applied.
 - ◇ That is, copy payments are automatically made: subtracted from the licensees account and forwarded to the licensor.
- ★ Although no reference is made to any edit functions these are made available to the licensee when invoking the edit command.
 - ◇ You may think of these edit functions being downloaded at the time of downloading the license.
 - ◇ Other than this we need not further specify the editing functions.
- ★ Same remarks apply to the above copying functions.

[**Domain Modelling**, **Scripts**, **The Performing Arts: Producers and Consumers**, **A License Language**]

- (10.) The licensee can further **sub-license** copied and/or edited work.
 - ★ The licensee must give the license being assembled a unique name.
 - ★ And the licensee must choose to whom to license this work.
 - ★ A sub-license, like does a license, authorises which actions can be performed, and then with which one of a set of alternative selection sequences.
 - ★ No payment function is explicitly mentioned.
 - ◇ It is to be semi-automatically derived
 - ◇ (from the originally licensed payment fee function
 - ◇ and the licensee's payment demands)
 - ◇ by means of functionalities provided as part of the licensed payment fee function.

[**Domain Modelling**, **Scripts**, **The Performing Arts: Producers and Consumers**, **A License Language**]

- ★ The sub-license command information is thus **compiled** (**assembled**) into a license of the form given in (1.–3.).
 - ◇ The licensee becomes the licensor and the recipient of the new, the sub-license, become the new licensee.
 - ◇ The assemblage refers to the context of the action.
 - ◇ That context knows who, the licensor, is issuing the sub-license.
 - ◇ From the license label of the command it is known whether the sub-license actions are a subset of those for which sub-licensing has been permitted.

Domain Modelling, Scripts

A Hospital Health Care License Language

- Citizens
 - ★ go to hospitals
 - ★ in order to be treated for some calamity (disease or other),
 - ★ and by doing so these citizens become patients.
- At hospitals patients, in a sense, issue a request to be treated with the aim of full or partial restitution.
- This request is directed at medical staff, that is,
 - ★ the patient authorises medical staff to perform a set of actions upon the patient.
 - ★ One could claim, as we shall, that the patient issues a license.

[Domain Modelling, Scripts, A Hospital Health Care License Language]

● Patients and Patient Medical Records ●

- So patients and their attendant patient medical records (PMRs) are the main entities, the “works” of this domain.
- We shall treat them synonymously: PMRs as surrogates for patients.
- Typical actions on patients — and hence on PMRs — involve
 - ★ admitting patients,
 - ★ interviewing patients,
 - ★ analysing patients,
 - ★ diagnosing patients,
 - ★ planning treatment for patients,
 - ★ actually treating patients, and,
 - ★ under normal circumstance, to finally release patients.

[Domain Modelling, Scripts, A Hospital Health Care License Language]

● Medical Staff ●

- Medical staff may request ('refer' to)
 - ★ other medical staff to perform some of these actions.
 - ★ One can conceive of describing action sequences (and 'referrals') in the form of hospitalisation (not treatment) plans.
 - ★ We shall call such scripts for licenses.

● Professional Health Care ●

- The issue is now,
 - ★ given that we record these licenses,
 - ★ their being issued and being honoured,
 - ★ whether the handling of patients at hospitals
 - ◇ follow,
 - ◇ or does not followproperly issued licenses.

[Domain Modelling, Scripts, A Hospital Health Care License Language]

● A Notion of License Execution State ●

- In the context of the Artistic License Language licensees could basically perform licensed actions in any sequence and as often as they so desired.
 - ★ There were, of course, some obvious constraints.
 - ◇ Operations on local works could not be done before these had been created — say by copying.
 - ◇ Editing could only be done on local works and hence required a prior action of, for example, copying a licensed work.
- In the context of hospital health care most of the actions can only be performed if the patient has reached a suitable state in the hospitalisation.
- We refer to Fig. 2.3 on the facing page for an idealised hospitalisation plan.

[Domain Modelling, Scripts, A Hospital Health Care License Language, A Notion of License Execution State]

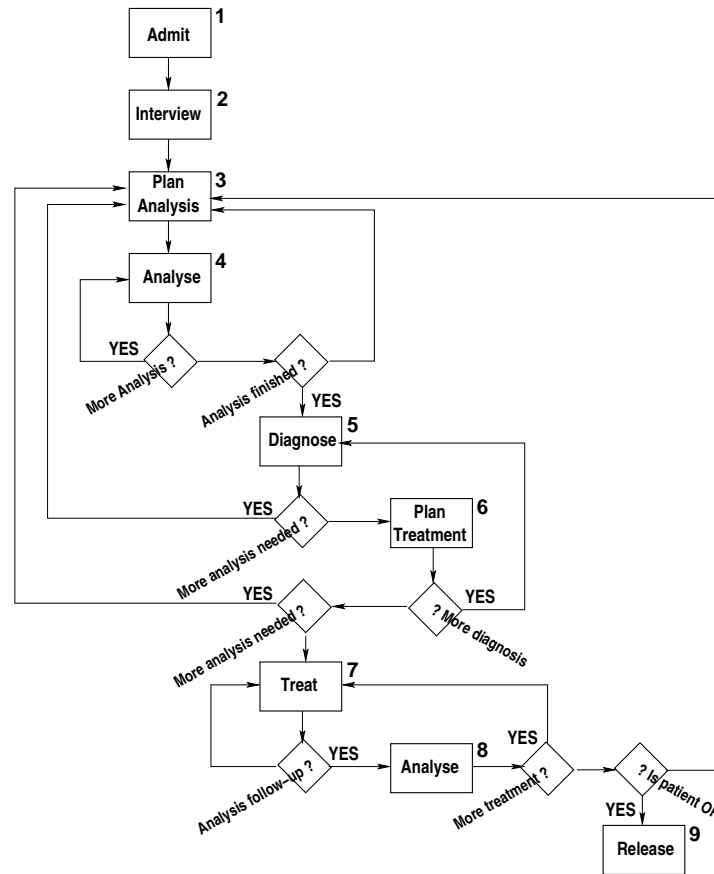


Figure 2.3: An example hospitalisation plan. States: {1,2,3,4,5,6,7,8,9}

[Domain Modelling, Scripts, A Hospital Health Care License Language, A Notion of License Execution State]

- We therefore suggest
 - ★ to join to the licensed commands
 - ★ an indicator which prescribe the (set of) state(s) of the hospitalisation plan in which the command action may be performed.
- Two or more medical staff may now be licensed
 - ★ to perform different (or even same !) actions
 - ★ in same or different states.
 - ★ If licensed to perform same action(s) in same state(s) —
 - ★ well that may be “bad license programming” if and only if it is bad medical practice !
- One cannot design a language and prevent it being misused!

[Domain Modelling, Scripts, A Hospital Health Care License Language]

● The License Language ●

- The syntax has two parts.
 - ★ One for licenses being issued by licensors.
 - ★ And one for the actions that licensees may wish to perform.

[Domain Modelling, Scripts, A Hospital Health Care License Language, The License Language]

type

0. Ln, Mn, Pn

1. License = Ln × Lic

2. Lic == mkLic(staff1:Mn,mandate:ML,pat:Pn)

3. ML == mkML(staff2:Mn,to_perform_acts:CoL-**set**)

4. CoL = Cmd | ML | Alt

5. Cmd == mkCmd(σ s: Σ -**set**,stmt:Stmt)

6. Alt == mkAlt(cmds:Cmd-**set**)

7. Stmt = **admit** | **interview** | **plan-analysis** | **do-analysis**
| **diagnose** | **plan-treatment** | **treat** | **transfer** | **release**

- The above syntax is correct RSL.
- But it is decorated!
- The subtypes {**|boldface keyword|**} are inserted for readability.

[Domain Modelling, Scripts, A Hospital Health Care License Language, The License Language]

- (0.) Licenses, medical staff and patients have names.
- (1.) Licenses further consist of license bodies (Lic).
- (2.) A license body names the licensee (Mn), the patient (Pn), and,
- (3.) through the “mandated” licence part (ML), it names the licensor (Mn) and which set of commands (C) or (o) implicit licenses (L, for CoL) the licensor is mandated to issue.
- (4.) An explicit command or licensing (CoL) is either a command (Cmd), or a sub-license (ML) or an alternative.
- (5.) A command (Cmd) is a state-labelled statement.

[Domain Modelling, Scripts, A Hospital Health Care License Language, The License Language]

- (3.) A sub-license just states the command set that the sub-license licenses.
 - ★ As for the Artistic License Language the licensee
 - ★ chooses an appropriate subset of commands.
 - ★ The context “inherits” the name of the patient.
 - ★ But the sub-licensee is explicitly mandated in the license!
- (6.) An alternative is also just a set of commands.
 - ★ The meaning is that
 - ◇ either the licensee choose to perform the designated actions
 - ◇ or, as for ML, but now freely choosing the sub-licensee,
 - ◇ the licensee (now new licensor) chooses to confer actions to other staff.

[Domain Modelling, Scripts, A Hospital Health Care License Language, The License Language]

- (7.) A statement is either
 - ★ an admit,
 - ★ an interview,
 - ★ a plan analysis,
 - ★ an analysis,
 - ★ a diagnose,
 - ★ a plan treatment,
 - ★ a treatment,
 - ★ a transfer, or
 - ★ a release
- directive
- Information given in the patient medical report
 - ★ for the designated state
 - ★ inform medical staff as to the details
 - ★ of analysis, what to base a diagnosis on, of treatment, etc.

[[Domain Modelling](#), [Scripts](#), [Licenses](#), A Hospital Health Care License Language, *The License Language*]

8. $\text{Action} = \text{Ln} \times \text{Act}$

9. $\text{Act} = \text{Stmt} \mid \text{SubLic}$

10. $\text{SubLic} = \text{mkSubLic}(\text{sublicensee}:\text{Ln}, \text{license}:\text{ML})$

[**Domain Modelling**, **Scripts**, **Hospital Health Care: Patients and Medical Staff**, **A Hospital Health Care License Language**]

- (8.) Each action actually attempted by a medical staff refers to the license, and hence the patient name.
- (9.) Actions are either of
 - ★ an admit,
 - ★ an interview,
 - ★ a plan analysis,
 - ★ an analysis,
 - ★ a diagnose,
 - ★ a plan treatment,
 - ★ a treatment,
 - ★ a transfer, or
 - ★ a releaseactions.

[Domain Modelling, Scripts, A Hospital Health Care License Language, The License Language]

- Each individual action is only allowed in a state σ
 - ★ if the action directive appears in the named license
 - ★ and the patient (medical record) designates state σ .
- (10.) Or an action can be a sub-licensing action.
 - ★ Either the sub-licensing action that the licensee is attempting is explicitly mandated by the license (4. ML),
 - ★ or is an alternative one thus implicitly mandated (6.).
 - ★ The full sub-license, as defined in (1.–3.) is compiled from contextual information.

Domain Modelling, Scripts

Public Government and the Citizens**• The Three Branches of Government •**

- By public government we shall,
 - ★ following Charles de Secondat, baron de Montesquieu (1689–1755),
 - ★ understand a composition of three powers:
 - ◇ the law-making (legislative),
 - ◇ the law-enforcing and
 - ◇ the law-interpretingparts of public government.

[Domain Modelling, Scripts, Public Government and the Citizens, The Three Branches of Government]

- Typically
 - ★ national parliament and local (province and city) councils are part of law-making government,
 - ★ law-enforcing government is called the executive (the administration),
 - ★ and law-interpreting government is called the judiciary [system] (including lawyers etc.).

[Domain Modelling, Scripts, Public Government and the Citizens]

• Documents •

- A crucial means of expressing public administration is through *documents*.
- We shall therefore provide a brief domain analysis of a concept of documents.
- (This document domain description also applies
 - ★ to patient medical records and,
 - ★ by some “light” interpretation, also to artistic works —insofar as they also are documents.)

[Domain Modelling, Scripts, Public Government and the Citizens, Documents]

- Documents are
 - ★ *created*,
 - ★ *edited* and
 - ★ *read*;
- and documents can be
 - ★ *copied*,
 - ★ *distributed*,
 - ★ the subject of *calculations* (interpretations) and be
 - ★ *shared* and
 - ★ *shredded*.

[Domain Modelling, Scripts, Public Government and the Citizens]

• Document Attributes •

- With documents one can associate, as attributes of documents, the *actors* who
 - ★ created,
 - ★ edited,
 - ★ read,
 - ★ copied,
 - ★ distributed
 - ◇ (to whom distributed),
 - ★ shared,
 - ★ performed calculations and
 - ★ shredded
- documents.

[Domain Modelling, Scripts, Public Government and the Citizens, Document Attributes]

- With these operations on documents,
 - and hence as attributes of documents one can, again conceptually,
 - associate the
 - ★ *location* and
 - ★ *time*
- of these operations.

[Domain Modelling, Scripts, Public Government and the Citizens]

● Actor Attributes and Licenses ●

- With actors (whether agents of public government or citizens)
 - ★ one can associate the *authority* (i.e., the *rights*)
 - ★ these actors have with respect to performing actions on documents.
- We now intend to express these *authorisations as licenses*.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#)]

● Document Tracing ●

- An issue of public government is
 - ★ whether citizens and agents of public government act in accordance with the laws —
 - ★ with actions and laws reflected in documents
 - ★ such that the action documents enables a trace from the actions to the laws “governing” these actions.

- We shall therefore assume that every document can be traced
 - ★ back to its law-origin
 - ★ as well as to all the documents any one document-creation or -editing was based on.

[Domain Modelling, Scripts, Public Government and the Citizens]

• A Document License Language •

- The syntax has two parts.
 - ★ One for licenses being issued by licensors.
 - ★ And one for the actions that licensees may wish to perform.

• The Form of Licenses •

type

0. Ln, An, Cfn

1. L == Grant | Extend | Restrict | Withdraw

2. Grant == mkG(license:Ln,licensor:An,granted_ops:Op-set,licensee:An)

3. Extend == mkE(licensor:An,licensee:An,license:Ln,with_ops:Op-set)

4. Restrict == mkR(licensor:An,licensee:An,license:Ln,to_ops:Op-set)

5. Withdraw == mkW(licensor:An,licensee:An,license:Ln)

6. Op == Crea|Edit|Read|Copy|Licn|Shar|Rvok|Rlea|Rtur|Calc|Shrd

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

type

7. Dn, DCn, UDI
8. Crea == mkCr(dn:Dn,doc_class:DCn,based_on:UDI-**set**)
9. Edit == mkEd(doc:UDI,based_on:UDI-**set**)
10. Read == mkRd(doc:UDI)
11. Copy == mkCp(doc:UDI)
- 12a. Licn == mkLi(kind:LiTy)
- 12b. LiTy == grant | extend | restrict | withdraw
13. Shar == mkSh(doc:UDI,with:An-**set**)
14. Rvok == mkRv(doc:UDI,from:An-**set**)
15. Rlea == mkRl(dn:Dn)
16. Rtur == mkRt(dn:Dn)
17. Calc == mkCa(fcts:CFn-**set**,docs:UDI-**set**)
18. Shrd == mkSh(doc:UDI)

[**Domain Modelling**, **Scripts**, **Public Government and the Citizens**, **A Document License Language**]

- (0.) There are names of licenses (L_n), actors (A_n), documents (UDI), document classes (DC_n) and calculation functions (C_{fn}).
- (1.) There are four kinds of licenses: granting, extending, restricting and withdrawing.
- (2.) Actors (licensors) grant licenses to other actors (licensees).
 - ★ An actor is constrained to always grant distinctly named licenses.
 - ★ No two actors grant identically named licenses.
 - ★ A set of operations on (named) documents are granted.

[Domain Modelling, Scripts, Public Government and the Citizens, A Document License Language]

- (3.–5.) Actors who have issued named licenses may extend, restrict or withdraw the license rights (wrt. operations, or fully).
- (6.) There are nine kinds of operation authorisations. Some of the next explications also explain parts of some of the corresponding actions (see (16.–24.)).
- (7.) There are names of documents (Dn), names of classes of documents (DCn), and there are unique document identifiers (UDI).

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (8.) **Creation** results in an initially void document which is
 - ★ not necessarily uniquely named (dn:Dn) (but that name is uniquely associated with the unique document identifier created when the document is created)
 - ★ typed by a document class name (dcn:DCn) and possibly
 - ★ based on one or more identified documents (over which the licensee (at least) has reading rights).
 - ★ We can presently omit consideration of the document class concept.
 - ★ “based on” means that the initially void document contains references to those (zero, one or more) documents.
 - ★ The “based on” documents are moved from licensor to licensee.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (9.) **Editing** a document

- ★ may be based on “inspiration” from, that is, with reference to a number of other documents (over which the licensee (at least) has reading rights).
- ★ What this “be based on” means is simply that the edited document contains those references. (They can therefore be traced.)
- ★ The “based on” documents are moved from licensor to licensee — if not already so moved as the result of the specification of other authorised actions.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (10.) **Reading** a document
 - ★ only changes its “having been read” status.
 - ★ The read document, if not the result of a copy, is moved from licensor to licensee — if not already so moved as the result of the specification of other authorised actions.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (11.) **Copying** a document
 - ★ increases the document population by exactly one document.
 - ★ All previously existing documents remain unchanged except that the document which served as a master for the copy has been so marked.
 - ★ The copied document is like the master document except that the copied document is marked to be a copy.
 - ★ The master document, if not the result of a create or copy, is moved from licensor to licensee
 - ★ — if not already so moved as the result of the specification of other authorised actions.

[Domain Modelling, Scripts, Public Government and the Citizens, A Document License Language]

- (12a.) A licensee can **sub-license** (sL) certain operations to be performed by other actors.
- (12b.) The granting, extending, restricting or withdrawing permissions,
 - ★ cannot name a license (the user has to do that),
 - ★ do not need to refer to the licensor (the licensee issuing the sub-license),
 - ★ and leaves it open to the licensor to freely choose a licensee.
 - ★ The licensor (the licensee issuing the sub-license) must choose a unique license name.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (13.) A document can be **shared**
 - ★ between two or more actors.
 - ★ One of these is the licensee, the others are implicitly given read authorisations.
 - ★ (One could think of extending, instead the licensing actions with a **shared** attribute.)
 - ★ The shared document, if not the result of a create and edit or copy, is moved from licensor to licensee — if not already so moved as the result of the specification of other authorised actions.
 - ★ Sharing a document does not move nor copy it.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (14.) Sharing documents can be **revoked**. That is, the reading rights are removed.
- (15.) The **release** operation:
 - ★ if a licensor has authorised a licensee to create a document
 - ★ (and that document, when created got the unique document identifier udi:UDI)
 - ★ then that licensee can **release** the created, and possibly edited document (by that identification)
 - ★ to the licensor, say, for comments.
 - ★ The licensor thus obtains the master copy.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (16.) The **return** operation:
 - ★ if a licensor has authorised a licensee to create a document
 - ★ (and that document, when created got the unique document identifier udi:UDI)
 - ★ then that licensee can **return** the created, and possibly edited document (by that identification)
 - ★ to the licensor — “for good”!
 - ★ The licensee relinquishes all control over that document.

[Domain Modelling, Scripts, Public Government and the Citizens, A Document License Language]

- (17.) Two or more documents can be subjected to any one of a set of permitted **calculation** functions.
 - ★ These documents, if not the result of a creates and edits or copies, are moved from licensor to licensee —
 - ★ if not already so moved as the result of the specification of other authorised actions.
 - ★ Observe that there can be many calculation permissions, over overlapping documents and functions.
- (18.) A document can be **shredded**.
 - ★ It seems pointless to shred a document if that was the only right granted wrt. document.

[**Domain Modelling**, **Scripts**, **Public Government and the Citizens**, **A Document License Language**]

17. Action = Ln × Clause

18. Clause = Cre | Edt | Rea | Cop | Lic | Sha | Rvk | Rel | Ret | Cal | Shr

19. Cre == mkCre(dcn:DCn,based_on_docs:UID-**set**)

20. Edt == mkEdt(uid:UID,based_on_docs:UID-**set**)

21. Rea == mkRea(uid:UID)

22. Cop == mkCop(uid:UID)

23. Lic == mkLic(license:L)

24. Sha == mkSha(uid:UID,with:An-**set**)

25. Rvk == mkRvk(uid:UID,from:An-**set**)

25. Rev == mkRev(uid:UID,from:An-**set**)

26. Rel == mkRel(dn:Dn,uid:UID)

27. Ret == mkRet(dn:Dn,uid:UID)

28. Cal == mkCal(fct:Cfn,over_docs:UID-**set**)

29. Shr == mkShr(uid:UID)

[Domain Modelling, Scripts, Public Government and the Citizens, A Document License Language]

- A clause elaborates to a state change and usually some value.
- The value yielded by elaboration of the above
 - ★ create, copy, and calculation
- clauses
- are **unique document identifiers**.
- These are chosen by the “system”.

[Domain Modelling, Scripts, Public Government and the Citizens, A Document License Language]

- (17.) Actions are **tagged** by the name of the license
 - ★ with respect to which their authorisation and document names has to be checked.
 - ★ No action can be performed by a licensee
 - ★ unless it is so authorised by the named license,
 - ★ both as concerns the operation (create, edit, read, copy, license, share, revoke, calculate and shred)
 - ★ and the documents actually named in the action.
 - ★ They must have been mentioned in the license,
 - ★ or, created or copies of downloaded (and possibly edited) documents or copies of these — in which cases operations are inherited.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (19.) A licensee may **create** documents if so licensed —
 - ★ and obtains all operation authorisations to this document.
- (20.) A licensee may **edit** “downloaded” (edited and/or copied) or created documents.
- (21.) A licensee may **read** “downloaded” (edited and/or copied) or created and edited documents.
- (22.) A licensee may (conditionally) **copy** “downloaded” (edited and/or copied) or created and edited documents.
 - ★ The licensee decides which name to give the new document, i.e., the copy.
 - ★ All rights of the master are inherited to the copy.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (23.) A licensee may **issue licenses**
 - ★ of the kind permitted.
 - ★ The licensee decides whether to do so or not.
 - ★ The licensee decides
 - ◇ to whom,
 - ◇ over which, if any, documents,
 - ◇ and for which operations.
 - ★ The licensee looks after a proper ordering of licensing commands:
 - ◇ first grant,
 - ◇ then sequences of zero, one or more either extensions or restrictions,
 - ◇ and finally, perhaps, a withdrawal.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (24.) A “downloaded” (possibly edited or copied) document may (conditionally) be **shared** with one or more other actors.
 - ★ Sharing, in a digital world, for example,
 - ★ means that any edits done after the opening of the sharing session,
 - ★ can be read by all so-granted other actors.
- (25.) Sharing may (conditionally) be **revoked**, partially or fully, that is, wrt. original “sharers”.

[[Domain Modelling](#), [Scripts](#), [Public Government and the Citizens](#), [A Document License Language](#)]

- (26.) A document may be **released**.
 - ★ It means that the licensor who originally requested
 - ★ a document (named dn:Dn) to be created
 - ★ now is being able to see the results —
 - ★ and is expected to comment on this document
 - ★ and eventually to re-license the licensee to further work.
- (27.) A document may be **returned**.
 - ★ It means that the licensor who originally requested
 - ★ a document (named dn:Dn) to be created
 - ★ is now given back the full control over this document.
 - ★ The licensee will no longer operate on it.

[Domain Modelling, Scripts, Public Government and the Citizens, A Document License Language]

- (28.) A license may (conditionally) apply any of a licensed set of **calculation functions**
 - ★ to “downloaded” (edited, copied, etc.) documents,
 - ★ or can (unconditionally) apply any of a licensed set of calculation functions
 - ★ to created (etc.) documents.
 - ★ The result of a calculation is a document.
 - ★ The licensee obtains all operation authorisations to this document (— as for created documents).
- (29.) A license may (conditionally) **shred** a “downloaded” (etc.) document.

Domain Modelling, Scripts

Discussion: Comparisons

- We have “designed”, or rather proposed three different kinds of license languages.
 - ★ In which ways are they “similar”,
 - ★ and in which ways are they “different”?
- Proper answers to these questions can only be given after we have formalised these languages.
 - ★ The comparisons can be properly founded on
 - ★ comparing the semantic values of these languages.

[Domain Modelling, Scripts, Discussion: Comparisons]

- But before we embark on such formalisations
 - ★ we need some informal comparisons
 - ★ so as to guide our formalisations.
- So we shall attempt such analysis now with the understanding that it is only a temporary one.

[Domain Modelling, Scripts, Discussion: Comparisons]

• Work Items •

- The **work items** of the **artistic** license language(s) are essentially “kept” by the licensor.
- The **work items** of the **hospital health care** license language(s) are fixed and, for a large set of licenses there is one work item, the patient which is shared between many licensors and licenses.
- The **work items** of the **public administration** license language(s) — namely document — are distributed to or created and copied by licenses and may possibly be shared.

[[Domain Modelling](#), [Scripts](#), [Discussion: Comparisons](#)]

• Operations •

- The **operations** of the **artistic** license language(s) are essentially “kept” by the licensor.
- The **operations** of the **hospital health care** license language(s) are essentially “kept” by the licensees (as reflected in their professional training and conduct).
- The **operations** of the **public administration** license language(s) are essentially “kept” by the licensees (as reflected in their professional training and conduct).

[[Domain Modelling](#), [Scripts](#), [Discussion: Comparisons](#)]

● Permissions and Obligations ●

- Generally we can say that
 - ★ the **modalities** of the **artistic** license language(s) are essentially **permissions** with **payment** (as well as use of licensor functions) being an **obligation**;
 - ★ that the **modalities** of the **hospital health care** license language(s) are essentially **obligations**;
 - ★ and, as well, that the **modalities** of the **public administration** license language(s) are essentially **obligations**
- We shall have more to say about permissions and obligations later (much later).

Domain Modelling, Scripts

Script and Contract Languages

- By a **domain script language** we mean
 - ★ the definition of a set of licenses and actions
 - ★ where these licenses when issued
 - ★ and actions when performed have morally obliging power.
- By a **domain contract language** we mean
 - ★ a domain script language whose licenses and actions
 - ★ have legally binding power, that is,
 - ◇ the issue of licenses and
 - ◇ the invocation of actions
 - may be contested in a court of law.
 - ★ We now refer to licenses as contracts.

Domain Modelling, Scripts

Review of Support Examples



- TO BE WRITTEN



[Domain Modelling, Scripts, Review of Support Examples]

● The Aircraft Simulator Script ●

-

- TO BE WRITTEN

-

[Domain Modelling, Scripts, Review of Support Examples]

● The Bill-of-Lading Script ●

-

- TO BE WRITTEN

-

[Domain Modelling, Scripts, Review of Support Examples]

- The Timetable Script Language •

-

- TO BE WRITTEN

-

[[Domain Modelling](#), [Scripts](#), [Review of Support Examples](#)]

- **The Bus Transport Contract Language** ●

-

- TO BE WRITTEN

-

[Domain Modelling, Scripts, Review of Support Examples, The Bus Transport Contract Language]



- TO BE WRITTEN



[Domain Modelling, Scripts, Review of Support Examples, The Bus Transport Contract Language]



- TO BE WRITTEN



Domain Modelling, Scripts

Modelling Scripts

Dines Bjørner: 8th DRAFT: October 14, 2008

[Domain Modelling]

Human Behaviours

Characterisation 70 (Human Behaviour) By **human behaviour** we mean

- any of a quality spectrum of carrying out assigned work:
 - ★ from **careful, diligent** and **accurate**,
 - via
 - ★ **sloppy** dispatch, and
 - ★ **delinquent** work,
 - to
 - ★ outright **criminal** pursuit



Domain Modelling, Human Behaviours

A Meta-characterisation of Human Behaviour

- Commensurate with the above, humans interpret rules and regulations differently,
- and not always consistently — in the sense of repeatedly applying the same interpretations.
- Our final specification pattern is therefore:

type

$$\text{Action} = \Theta \rightsquigarrow \Theta\text{-infset}$$
value

$$\text{hum_int}: \text{Rule} \rightarrow \Theta \rightarrow \text{RUL-infset}$$

$$\text{action}: \text{Stimulus} \rightarrow \Theta \rightarrow \Theta$$

$$\text{hum_beha}: \text{Stimulus} \times \text{Rules} \rightarrow \text{Action} \rightarrow \Theta \rightsquigarrow \Theta\text{-infset}$$

$$\text{hum_beha}(\text{sy_sti}, \text{sy_rul})(\alpha)(\theta) \text{ as } \theta\text{set}$$
post

$$\theta\text{set} = \alpha(\theta) \wedge \text{action}(\text{sy_sti})(\theta) \in \theta\text{set}$$

$$\wedge \forall \theta': \Theta \cdot \theta' \in \theta\text{set} \Rightarrow$$

$$\exists \text{se_rul}: \text{RUL} \cdot \text{se_rul} \in \text{hum_int}(\text{sy_rul})(\theta) \Rightarrow \text{se_rul}(\theta, \theta')$$

[Domain Modelling, Human Behaviours, A Meta-characterisation of Human Behaviour]

- The above is, necessarily, sketchy:
 - ★ There is a possibly infinite variety of ways of interpreting some rules.
 - ★ A human, in carrying out an action, interprets applicable rules and chooses one which that person believes suits some (professional, sloppy, delinquent or criminal) intent.
 - ★ “Suits” means that it satisfies the intent,
 - ◇ i.e., yields **true** on the pre/post-configuration pair,
 - ◇ when the action is performed —
 - ◇ whether as intended by the ones who issued the rules and regulations or not.
 - ★ We do not cover the case of whether an appropriate regulation is applied or not.

[Domain Modelling, Human Behaviours, A Meta-characterisation of Human Behaviour]

- The above-stated axioms express how it is in the domain,
- not how we would like it to be.
- For that we have to establish requirements.

Domain Modelling, Human Behaviours

Review of Support Examples



- TO BE WRITTEN



[Domain Modelling, Human Behaviours, Review of Support Examples]



- TO BE WRITTEN



Domain Modelling, Human Behaviours

On Modelling Human Behaviour

- To model human behaviour is, “initially”, much like modelling management and organisation.
- But only ‘initially’.
- The most significant human behaviour modelling aspect is then that of modelling non-determinism and looseness, even ambiguity.
- So a specification language which allows specifying non-determinism and looseness (like CafeOBJ and RSL) is to be preferred.

[Domain Modelling]

Consolidation of Domain Description

[Domain Modelling]

Discussion of Facets

End of Lecture 8
